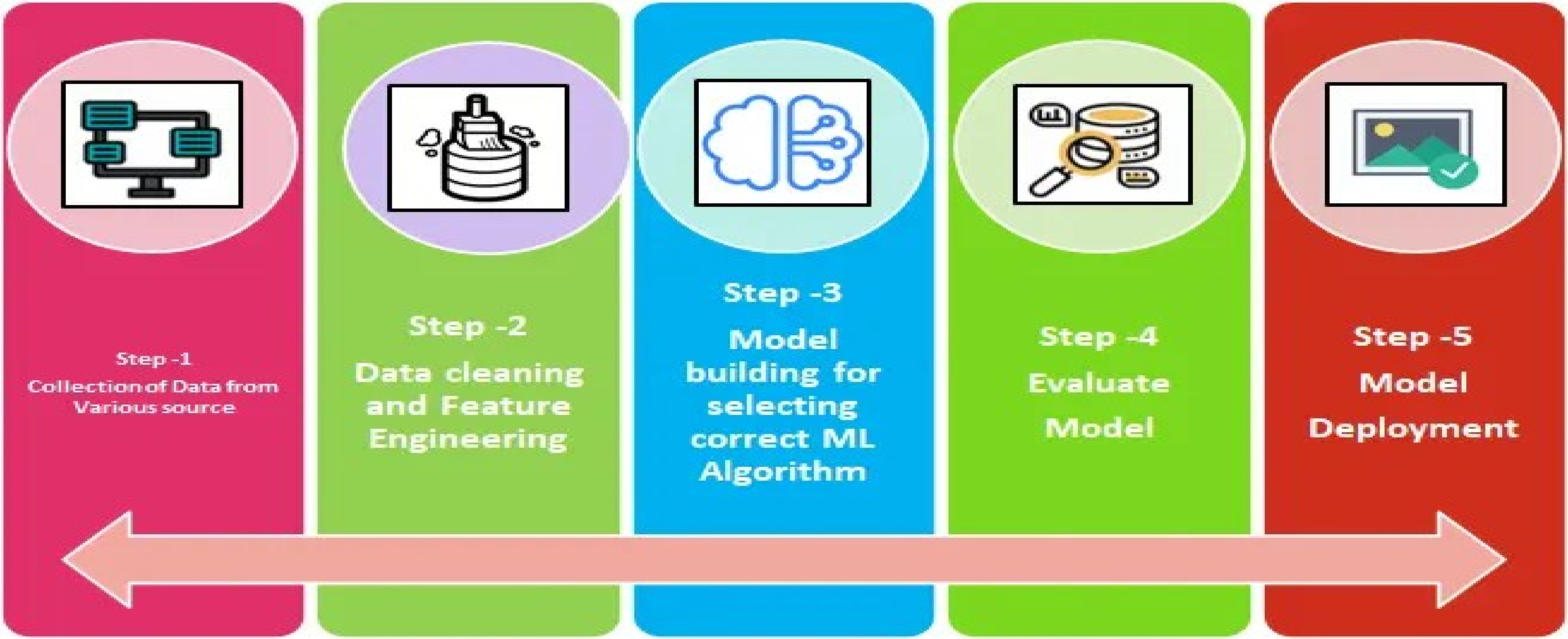
Chapter-4 Machine Learning- Supervised Learning

Basic Steps In ML

1. **Data collection**
   "training data", **mostly** with "labels" provided by a "teacher";
2. **Data preprocesing**
   Clean data to have homogenity
3. **Feature engineering**
   Select represenatative features to improve performance
4. **Modeling**
   choose the class of models that can describe the data
5. **Estimation/Selection**
   find the model that best explains the data: simple and fits well;
6. **Validation**
   evaluate the learned model and compare to solution found using other model classes;
7. **Operation**
   Apply learned model to new "test" data or real world instances

Basic Steps In ML



Common Terms

Features and Labels:

Features: These are the **input variables or characteristics** that the machine learning algorithm uses to make predictions.

Features provide the information on which the model's predictions are based.

The quality and relevance of features significantly **impact the performance of the machine learning model.**

House Price Prediction: Square footage, number of bedrooms, location,

number of bathrooms, presence of a garage.

Email Spam Classification: Email content, sender's address, presence of certain keywords.

Image Classification: Pixel values of an image, color distribution, texture features.

Common Terms

▪ **Labels,** also known as the **target variable or output variable,** represent the desired **outcome or prediction** that the model aims to achieve.

• Labels are the values that the model is **trying to predict.**

• The model's performance is assessed based on how well it predicts or approximates these labels.

House Price Prediction: Label: The actual price of the house.

Email Spam Classification: Label: Spam or not spam.

Image Classification: Label: Object categories (e.g. cat, dog, car).

Common Terms

- **Training Data:**

The training data is a subset of the available dataset that is used to train the machine learning model

- During training, the model adjusts its parameters based on this data to make accurate predictions.

Common Terms

- **Testing Data:**

- Once the model is trained on the training data, it is evaluated on a separate subset of data that was not used during the training process

- This testing data allows assessing how well the model generalizes to new, unseen data. Provides an unbiased evaluation of the model's ability to generalize.

- Helps identify if the model has **overfitting or underfitting** to the training data and whether it can make accurate predictions on real-world examples.

Common Terms

- **Overfitting**

- This phenomenon occurs when a model performs **really well on the data that we used to train** it but it **fails to generalise** well to new, unseen data. due to noise , and the model learned to predict specific inputs rather than the predictive parameters helps to make correct predictions

Under-fitting

- the model has **poor performance even on the data** that was used to **train it.** In most cases, underfitting occurs because the model is **not suitable for the problem** you are trying to solve

Data set Preparation

- To overcome over and under fitting try d/t approach of splitting

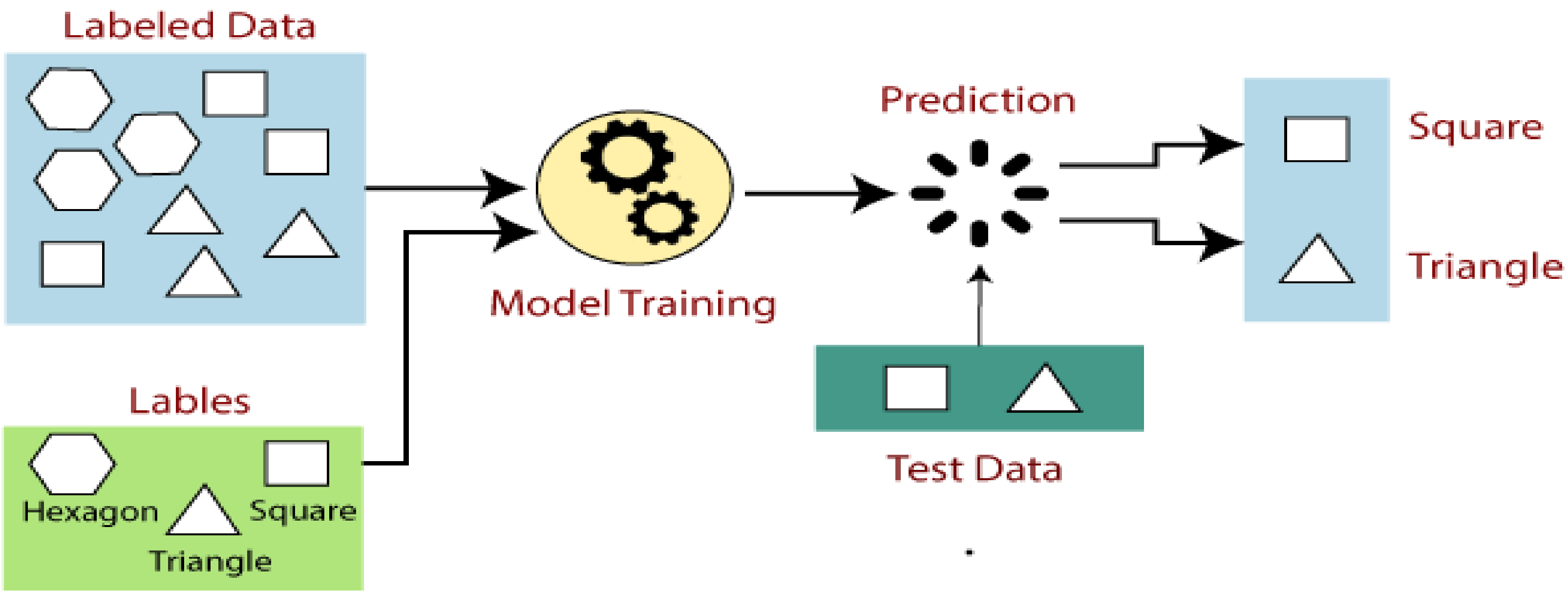
- simplest way to split the modelling dataset into **training and testing sets** is to assign **two thirds of the data** for training and rest for testing

Supervised ML

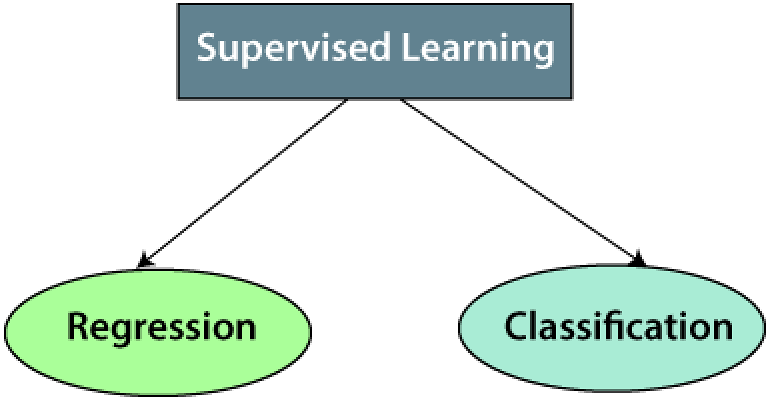
Supervised learning involves training an algorithm on a labeled dataset, where input data is paired with corresponding output labels.

- The goal is to learn a mapping from input to output based on provided labelled examples.

Supervised Learning



Supervised ML algorithms



Regression

Regression algorithms are used if there is a relationship between the input variable and the output variable.

- It is used for the prediction of **continuous variables**, such as Weather forecasting, Market Trends, etc. Below are some popular Regression algorithms which come under supervised learning: examples
  - Linear Regression
  - Non-Linear Regression
  - Polynomial Regression

Linear Regression

▪It is one of the very simple and easy algorithms which works on regression and shows the relationship between the continuous variables.

We should know that regression is a statistical method. It is used in finding relationships between variables.

▪Linear regression is one of the regression-based algorithms in ML. It shows a linear relationship between its variables.

Assume some company x spent the following cost for advertisement and get sale values as indicated ?

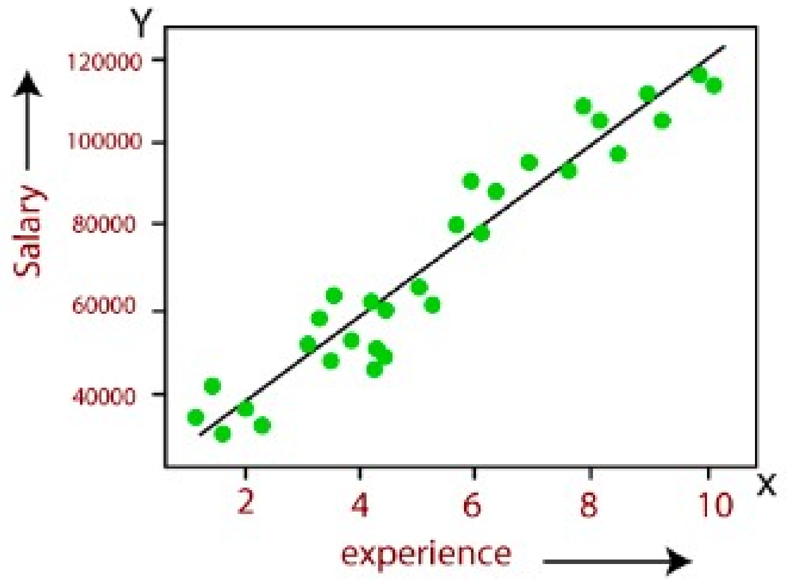
the company wants to do the advertisement of $200 in the year 2019 and wants to know the prediction about the sales for this year.

Example :

Advertisement	Sales
$90	$1000
$120	$1300
$150	$1800
$100	$1200
$130	$1380
$200	??

Linear reg cont…

2.Here we are predicting the **salary of an employee** on the basis of the year of experience.

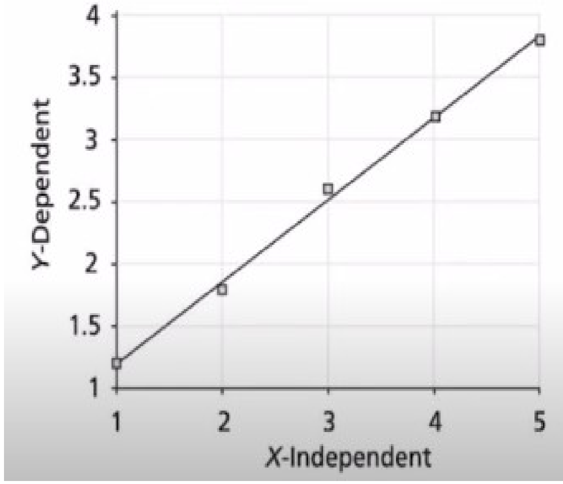


LR problem cont..

- Let us consider an example where the five weeks' sales data (in Thousands) is given as shown in Table.
- Apply linear regression technique to predict the 7th and 12th week sales.

x_i (Week)	y_j (Sales in Thousands)
1	1.2
2	1.8
3	2.6
4	3.2
5	3.8

Just finding the best fitting line



x_i (Week)	y_j (Sales in Thousands
1	1.2
2	1.8
3	2.6
4	3.2
5	3.8

Formula

$$y = \alpha + \beta x$$

β = slope

α = y-intercept

y = y- coordinate

x = x-coordinate

- Linear regression equation is given by
- $y = a_0 + a_1 * x + e$

- *where*
- $a_1 = \frac{(\overline{xy}) - (\bar{x})(\bar{y})}{\overline{x^2} - \bar{x}^2}$
- $a_0 = \bar{y} - a_1 * \bar{x}$

So/n cont...

- Here, there are 5 items, i.e., i = 1, 2, 3, 4, 5.

	x_i (Week)	y_j (Sales in Thousands)	x_i^2	$x_i * y_j$
	1	1.2	1	1.2
	2	1.8	4	3.6
	3	2.6	9	7.8
	4	3.2	16	12.8
	5	3.8	25	19
Sum	15	12.6	55	44.4
Average	$\bar{x} = 3$	$\bar{y} = 2.52$	$\overline{x^2} = 11$	$\overline{xy} = 8.88$

- *where*
- $a_1 = \frac{(\overline{xy}) - (\bar{x})(\bar{y})}{\overline{x^2} - \bar{x}^2}$
- $a_0 = \bar{y} - a_1 * \bar{x}$

Get correct regression line

- $\bar{x} = 3$ $\quad \bar{y} = 2.52$ $\quad \overline{x^2} = 11$ $\quad \overline{xy} = 8.88$
- $a_1 = \frac{(\overline{xy}) - (\bar{x})(\bar{y})}{\overline{x^2} - \bar{x}^2} = \frac{8.88 - 3*2.52}{11 - 3^2} = 0.66$
- $a_0 = \bar{y} - a_1 * \bar{x} = 2.52 - 0.66 * 3 = 0.54$
- **Regression equation is**
- $\boldsymbol{y = a_0 + a_1 * x}$
- $\boldsymbol{y = 0.54 + 0.66 * x}$

Linear Regression

- **Regression equation is**
- $y = a_0 + a_1 * x$
- $y = 0.54 + 0.66 * x$
- The predicted 7th week sale (when x = 7) is,
- **y = 0.54 + 0.66 x 7 = 5.16**
- the predicted 12th week sale (when x = 12) is,

Practical ML-Prediction problem

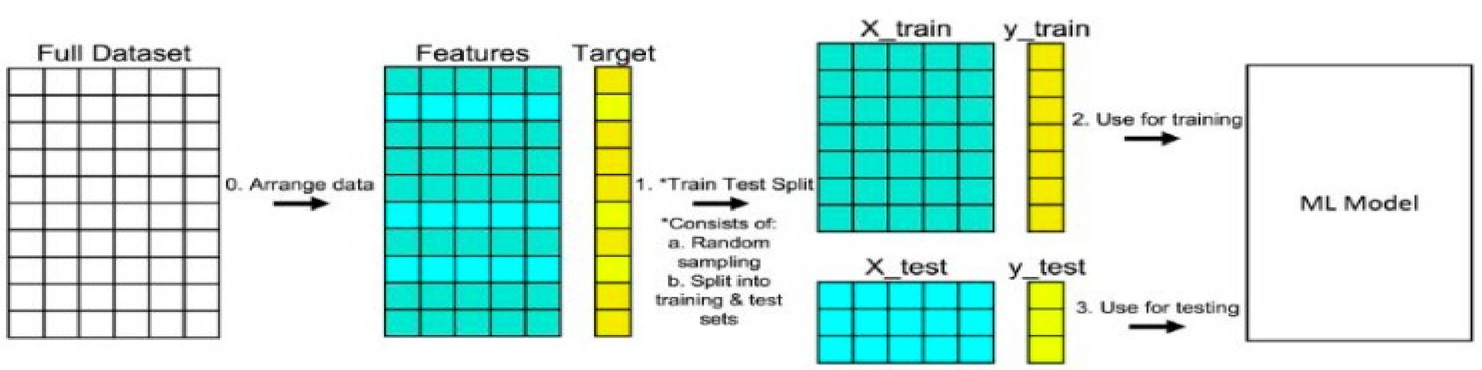
Step-1: Data Loading

Step 2:Identify Independent /Dependent variables */predictions*

Step 3: Split the data to train/test the ML algorithms

Step-4:train the model and test it

Train test split is a model validation procedure that allows you to simulate how a model would perform on new/unseen data. Here is how the procedure works:



Change this to practical ML

```
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0, train_size =
.75)
```

The random_state:

is a pseudo-random number parameter that allows you to **reproduce the same train test split** each time you run the code.

Select data set randomly before splitting just put / shuffling +ve integer commonly 42 , 2 ,0 default None

=unless you put random state you will get d/t values

Exercise : please split the following data with random state

X=[10,20,30,40,50,60,80,90,100]

Y=[1,0,1,4,5,6,7,8,9,10]

Sample Splitting task

```
from sklearn.model_selection import train_test_split
x=[10,20,30,40,50,60,80,90,100,200]
y=[1,0,1,4,5,6,7,8,9,10]
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=2)
print("x_train",x_train)
print("x_test",x_test)
print("y_train",y_train)
print("y_test",y_test)
```

Classification Algorithms

Classification algorithms are used when the **output variable is categorical**, which means there are two classes such as Yes-No, Male-Female, True-false,

Application of classification algorithms

- Email Spam Detection
- Speech Recognition
- Identifications of Cancer cases
- Drugs Classification
- Biometric Identification, etc.

Classification Algorithms

Classification algorithms are used when the **output variable is categorical**, which means there are two classes such as Yes-No, Male-Female, True-false,

Popular classification algorithms

- Logistic Regression
- Support vector Machines
- KNN
- Random Forest
- Decision Trees

Classification Algorithms

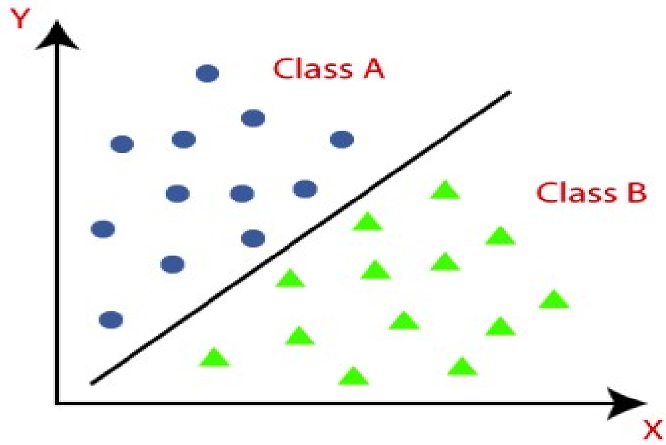
Classification algorithms are used when the **output variable is categorical,**

a program learns from the given **dataset or observations** and then classifies new observation into a number of classes or groups

Examples yes/no 0/1 True/False

- In classification algorithm, a discrete output function(y) is mapped to input variable(x).

  y=f(x), where y = categorical output



Classification Algorithms

- The algorithm which implements the classification on a dataset is known as a classifier.

There are two types of Classifications:

- **Binary Classifier:** classification problem has only two possible outcomes, then it is called as Binary Classifier.

Examples: YES or NO, MALE or FEMALE, SPAM or NOT SPAM, CAT or DOG, etc.

- **Multi-class Classifier:** If a classification problem has more than two outcomes, then it is called as Multi-class Classifier.

Example: Classifications of types of crops, Classification of types of music.

Types of ML Classification Algorithms

Classification Algorithms can be further divided into the Mainly two category:

Linear Models

- ✓ Logistic Regression
- ✓ Support Vector Machines

Non-linear Models

- ✓ K-Nearest Neighbours
- ✓ Decision Tree
- ✓ Naïve Bayes
- ✓ Random Forest

Evaluating a Classification model:

Evaluate the performance of Classification or Regression model.

1. Log Loss or Cross-Entropy Loss

- It is used for evaluating the performance of a classifier, whose output is a probability value between the 0 and 1.
- For a good binary Classification model, the value of log loss should be near to 0.
- The lower log loss represents the higher accuracy of the model.

2. Confusion Matrix

The confusion matrix provides us a matrix/table as output and describes the performance of the model.

It is also known as the error matrix.

Evaluating a Classification model:

The matrix consists of predictions result in a summarized form, which has a total number of correct predictions and incorrect predictions.

The matrix looks like as below table:

	Actual Positive	Actual Negative
Predicted Positive	True Positive	False Positive
Predicted Negative	False Negative	True Negative

$$\text{Accuracy} = \frac{TP+TN}{\text{Total Population}}$$

1. **True negatives:** correctly predicted negatives (zeros)
2. **True positives:** correctly predicted positives (ones)
3. **False negatives:** incorrectly predicted negatives (zeros)
4. **False positives:** incorrectly predicted positives (ones)

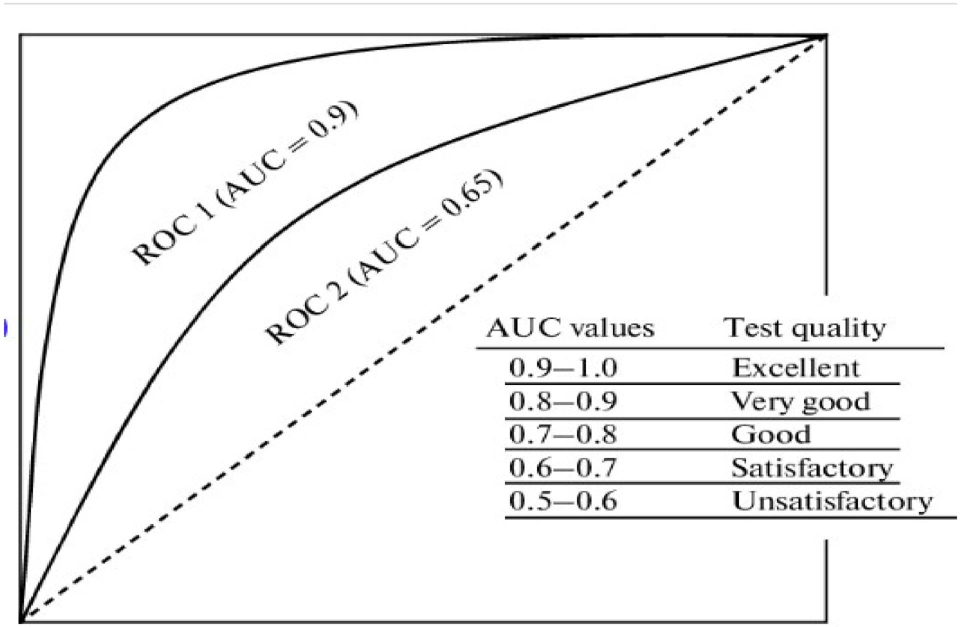
Evaluating a Classification model:

3.AUC-ROC curve:

- ROC curve stands for Receiver Operating Characteristics Curve and AUC stands for Area Under the Curve.
- It is a graph that shows the performance of the classification model at different thresholds.
- To visualize the performance of the multi-class classification model, we use the AUC-ROC Curve.
- The ROC curve is plotted with TPR and FPR, where TPR (True Positive Rate) on Y-axis and FPR(False Positive Rate) on X-axis.

Evaluating a Classification model Performance

AUC-ROC curve:



Evaluating a Classification model:

4. Precision, Recall, and F1-Score

These metrics are particularly useful in **binary or multiclass** classification.

Precision: The ratio of correctly **predicted positive observations** to the total predicted positives.

Recall: The ratio of **correctly predicted positive** observations to **all actual positives.**

F1-Score: The harmonic mean of **precision and recall.**

$$
\begin{aligned}
precision &= \frac{TP}{TP + FP} \\
recall &= \frac{TP}{TP + FN} \\
F1 &= \frac{2 \times precision \times recall}{precision + recall} \\
accuracy &= \frac{TP + TN}{TP + FN + TN + FP}
\end{aligned}
$$

Evaluating a Classification model:

Performance Evaluation for Regression Model

2. **Regression Metrics:**

Mean Absolute Error (MAE): The average **absolute differences** between **predicted and actual values.**

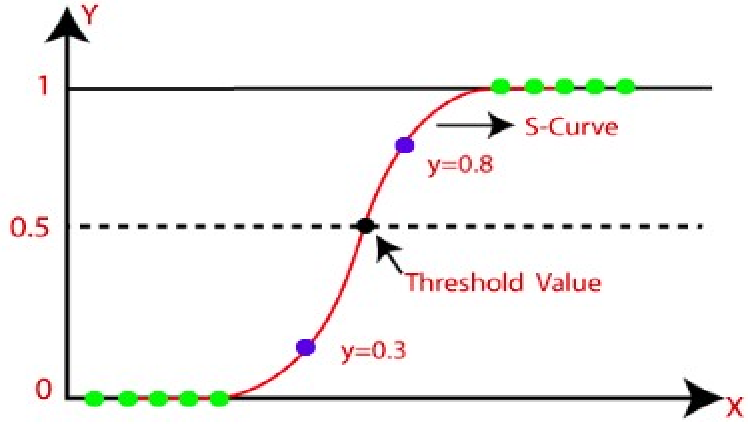
Mean Squared Error (MSE): The average of the squared differences between **predicted and actual values.**

Root Mean Squared Error (RMSE): The **square root of the MSE,** providing an interpretable scale.

Logistic Regression

- Logistic regression is one of the most popular Machine Learning algorithms
- Therefore the outcome must be a **categorical or discrete value.** It can be either Yes or No, 0 or 1, true or False, etc.
- it gives **the probabilistic** values which lie **between 0 and 1**.

In Logistic regression, instead of fitting a regression line, we fit **an "S" shaped logistic function**, which predicts two maximum values (0 or 1).



The S-form curve is called the Sigmoid function or the logistic function.

Types of Logistic Regression

- On the basis of the categories, Logistic Regression can be classified into three types:
- **Binomial**: In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.
- **Multinomial**: In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as "cat", "dogs", or "sheep"
- **Ordinal**: In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as "low", "Medium", or "High".

Python Implementation of Logistic Regression (Binomial)

- Data Pre-processing step
- Fitting Logistic Regression to the Training set
- Predicting the test result
- Test accuracy of the result(Creation of Confusion matrix)

Code samples

```
# Importing required libraries
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix

# Loading and preparing the data
data = pd.read_csv('data.csv')
X = data[['feature1', 'feature2', '...']]  # Selecting predictor variables
y = data['outcome']  # Selecting outcome variable

# Splitting the data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Creating and fitting the Logistic Regression model
model = LogisticRegression()
model.fit(X_train, y_train)

# Making predictions on test set
y_pred = model.predict(X_test)

# Evaluating the model
accuracy = accuracy_score(y_test, y_pred)
confusion = confusion_matrix(y_test, y_pred)

# Printing the results
print(f'Accuracy: {accuracy}')
print(f'Confusion Matrix: {confusion}')
```

Example -

There is a car making company that has recently launched a new car

So the company **wanted to check predict whether a user will purchase the product or not, one needs to find out the relationship between Age and Estimated Salary.**

User ID	Gender	Age	EstimatedSalary	Purchased
15624510	Male	19	19000	0
15810944	Male	35	20000	0
15668575	Female	26	43000	0
15603246	Female	27	57000	0
15804002	Male	19	76000	0
15728773	Male	27	58000	0
15598044	Female	27	84000	0
15694829	Female	32	150000	1
15600575	Male	25	33000	0
15727311	Female	35	65000	0
15570769	Female	26	80000	0
15606274	Female	26	52000	0
15746139	Male	20	86000	0
15704987	Male	32	18000	0
15628972	Male	18	82000	0
15697686	Male	29	80000	0
15733883	Male	47	25000	1
15617482	Male	45	26000	1
15704583	Male	46	28000	1
15621083	Female	48	29000	1
15649487	Male	45	22000	1
15736760	Female	47	49000	1

Source of data

https://www.kaggle.com/code/sandragracenelson/logistic-regression-on-user-data-csv/input

Data Processing -Related to our data set

Data Preprocessing Techniques Techniques you should apply:

1. How about if we want to include the **age as independent** variable

   Replace male and female with discrete values b/n **0** and **1**

2. as we see there is a variation b/n **age and salary** value which may create bias

So, need to apply **Feature scaling**

- **Feature scaling** is a method used to **normalize the range of independent variables** or features of data.
- it is also known as **data normalization** and is generally performed during the **data preprocessing step.**
- improve **model performance**, reduce the impact of outliers, and ensure that the data is on the same scale

2. K-Nearest Neighbors(KNN)

K-Nearest Neighbors (KNN) is a **simple and versatile** machine learning algorithm used for both **classification and regression** tasks.

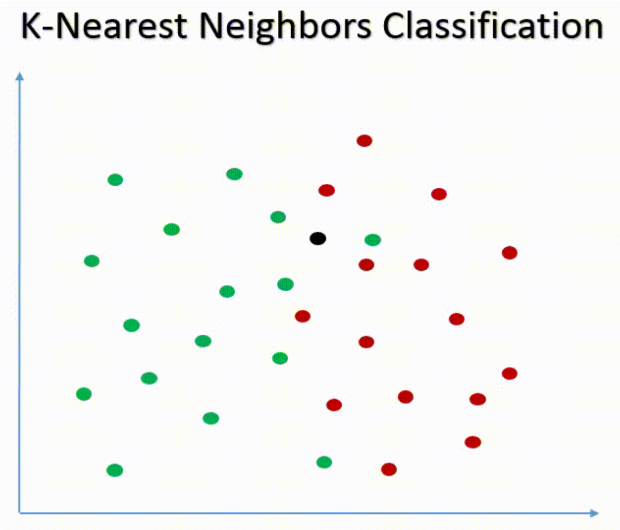
The fundamental idea behind KNN is to predict the label of a data point by **looking at its k nearest neighbors** in the feature space.

Technique to classify

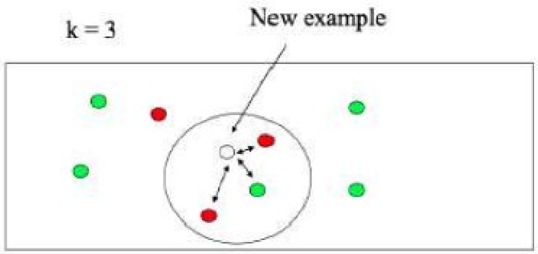
Given a new, unseen data point, find the **k-nearest neighbors** in the training set based on **some distance metric (Euclidean distance).**

For classification: Assign the **majority class label among** the k-nearest neighbors to the **new data point.**

K-Nearest Neighbors(KNN)



When unknown tuple is given – searches the pattern space for the k training tuples (k nearest nieghbors) that are closest to the unknown tuple.



- The most used distance function is the Euclidian distance:

$$d(X,Y)=\sqrt{\sum_{i=1}^{n}(x_i-y_i)^2}$$

K-Nearest Neighbors(KNN)

Advantages

– Conceptually simple, easy to understand and explain

– Very flexible decision boundaries

– Not much learning at all

Disadvantages

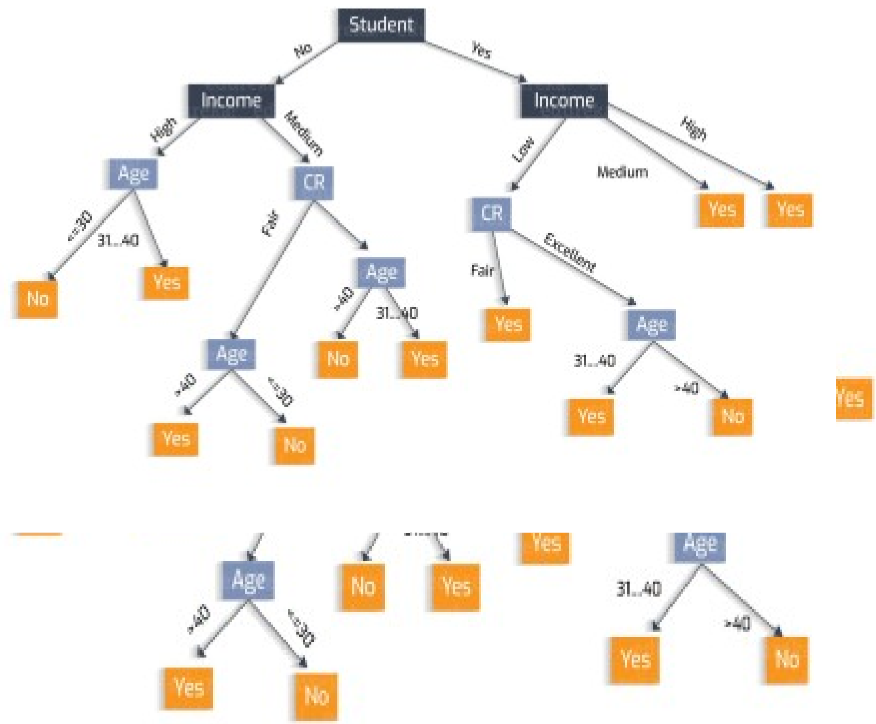
– It can be hard to find a good distance measure

– Irrelevant features and noise can be very detrimental

– Typically can not handle more than a few dozen attributes

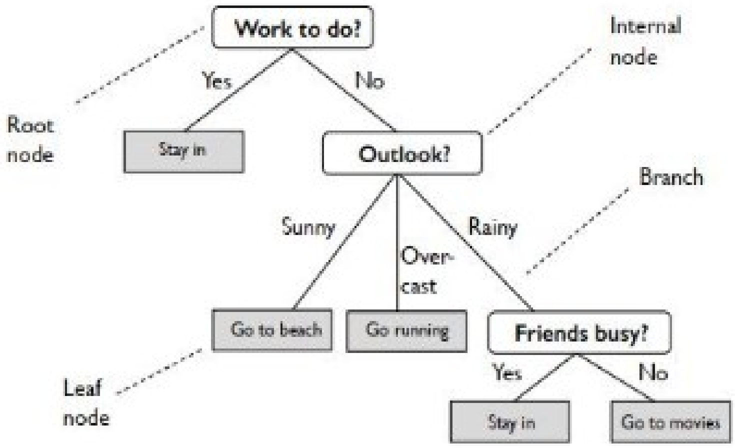
– Computational cost: requires a lot computation and memory

Decision Tree



Decision Tree

Decision Tree



Example of a non-binary decision tree with categorical features.

Decision tree structure

- **Root node:** beginning of a tree and represents **entire population** being analyzed.
- **Internal node:** denotes a test on an **attribute**
- **Branch:** represents an **outcome** of the test
- **Leaf nodes:** represent **class labels** or class distribution

- Tree is constructed in a top-down recursive divide-and-conquer manner

Decision Tree

Solving the classification problem using DT is a two-step process:

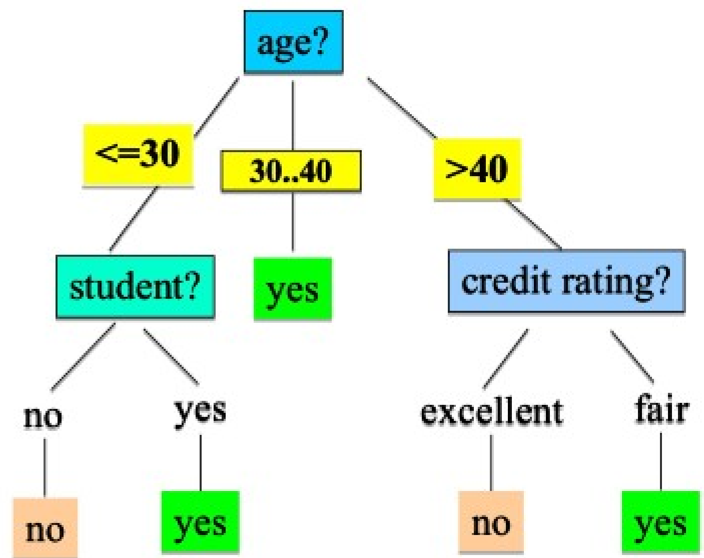
- **Decision Tree Induction-** Construct a DT using training data/Induction

Output: A Decision Tree for "*buys_computer*"

Training Dataset

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Decision Tree



Decision Tree-Algorithm

Classification Rule Extraction

- Represent the knowledge in the form of IF-THEN rules
- One rule is created for each path from the root to a leaf
- Each attribute-value pair along a path forms a conjunction
- The leaf node holds the class prediction
- Rules are easier for humans to understand
- Example

IF *age* = "<=30" AND *student* = "*no*" THEN *buys_computer* = "*no*"
IF *age* = "<=30" AND *student* = "*yes*" THEN *buys_computer* = "*yes*"
IF *age* = "31…40" THEN *buys_computer* = "*yes*"
IF *age* = ">40" AND *credit_rating* = "*excellent*" THEN *buys_computer* = "*yes*"
IF *age* = ">40" AND *credit_rating* = "*fair*" THEN *buys_computer* = "*no*"

Decision Tree-Python Implementation

Step : Import Library and Train the data

```
From sklearn.tree import DecisionTreeClassifier

classifier= DecisionTreeClassifier()
```

Key Term

Entropy

- Entropy is a measure of **impurity or disorder in a set.** In the context of decision trees, entropy is used to **calculate the information gain at each node.**

Gini impurity

- is another measure of impurity used in decision trees. I**t measures the likelihood of an incorrect classification of a randomly chosen element** if it were randomly labelled

Naïve Bayes ML Algorithm

- Naïve Bayes Classifier is one of the **simplest and most effective** Classification algorithms which helps in building the **fast machine learning models** that can make **quick predictions**.
- It is mainly used in **text classification** that includes a high-dimensional training dataset.
- Some popular examples of Naïve Bayes Algorithm are **spam filtration, Sentimental analysis, and classifying articles**.

Naïve Bayes ML Algorithm

- Bayesian classifiers are statistical classifiers.
- They can predict class membership probabilities, such as **the probability** that a **given tuple belongs** to a particular class.

- Bayesian classification is based on **Bayes' theorem**

$$P(h|X) = \frac{P(X|h)P(h)}{P(X)}$$

where h is hypothesis, X is a training data

- Compared to Decision tree, Byesian classifiers have also exhibited high accuracy and speed when applied to large databases.

Bayes Theorem

- Let X be a data tuple.
- Let H be some hypothesis such as that the data tuple X belongs to a specified class C.
- We want to determine P(H|X)– posterior probability of H conditioned on X
- We are looking for the probability that tuple X belongs to class C, given that we know the attribute description of X.

$$P(H|X) = \frac{P(X|H)P(H)}{P(X)}$$

Naïve Bayes ML Algorithm

Bayes theorem provides a way of computing posterior probability P(c|x) from P(c), P(x) and P(x|c). Look at the equation below:

$$\underbrace{P(c \mid x)}_{\text{Posterior Probability}} = \frac{\overbrace{P(x \mid c)}^{\text{Likelihood}}\,\overbrace{P(c)}^{\text{Class Prior Probability}}}{\underbrace{P(x)}_{\text{Predictor Prior Probability}}}$$

$$P(\text{Class}|\text{Features}) = \frac{P(\text{Features}|\text{Class}) \times P(\text{Class})}{P(\text{Features})}$$

$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

Above,

- *P*(*c*/*x*) is the posterior probability of *class* (c, *target*) given *predictor* (x, *attributes*).
- *P*(*c*) is the prior probability of *class*.
- *P*(*x*/*c*) is the likelihood which is the probability of the *predictor* given *class*.
- *P*(*x*) is the prior probability of the *predictor*.

Example : Naïve Bayes ML

Problem: using the given data set , classify or predict weather a person with the given condition will play tennis or not?

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

$\langle Outlook = sunny, Temperature = cool, Humidity = high, Wind = strong \rangle$

Example : Naïve Bayes ML

Step-1 calculate the prior/class label probability for Yes / No conditions Yes appeared 9 , and no appeared 5 out of 14 probability

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

$$P(PlayTennis = yes) = 9/14 = .64$$

$$P(PlayTennis = no) = 5/14 = .36$$

Example : Naïve Bayes ML

Step-2 calculate the conditional probability of individual attributes/predictors(outlook , temperature,Humidity,Windy)

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

$$P(PlayTennis = yes) = 9/14 = .64$$

$$P(PlayTennis = no) = 5/14 = .36$$

Outlook	Y	N
sunny	2/9	3/5
overcast	4/9	0
rain	3/9	2/5
Tempreature		
hot	2/9	2/5
mild	4/9	2/5
cool	3/9	1/5

Humidity	Y	N
high	3/9	4/5
normal	6/9	1/5
Windy		
Strong	3/9	3/5
Weak	6/9	2/5

Example : Naïve Bayes ML

Step-3 apply naive bayes formula to find new instance classification: sum up yes_ conditional probabilities of all feature and no probabilities , then compare the value lastly normalize it

$$\langle Outlook = sunny, Temperature = cool, Humidity = high, Wind = strong \rangle$$

Outlook	Y	N
sunny	2/9	3/5
overcast	4/9	0
rain	3/9	2/5
Tempreature		
hot	2/9	2/5
mild	4/9	2/5
cool	3/9	1/5

Humidity	Y	N
high	3/9	4/5
normal	6/9	1/5
Windy		
Strong	3/9	3/5
Weak	6/9	2/5

$$P(PlayTennis = yes) = 9/14 = .64$$

$$P(PlayTennis = no) = 5/14 = .36$$

$$v_{NB}(yes) = P(yes)\ P(sunny|yes)\ P(cool|yes)\ P(high|yes)\ P(strong|yes) = .0053$$

$$v_{NB}(no) = P(no)\ P(sunny|no)\ P(cool|no)\ P(high|no)\ P(strong|no) = .0206$$

$$v_{NB}(yes) = \frac{v_{NB}(yes)}{v_{NB}(yes)+v_{NB}(no)} = 0.205 \qquad v_{NB}(no) = \frac{v_{NB}(no)}{v_{NB}(yes)+v_{NB}(no)} = 0.795$$

Finaly , we can conclude that with the given features person **will not play tennis**

Example : Naïve Bayes ML

Step-3 based on the following classify the new species ?

No	Color	Legs	Height	Smelly	Species
1	White	3	Short	Yes	M
2	Green	2	Tall	No	M
3	Green	3	Short	Yes	M
4	White	3	Short	Yes	M
5	Green	2	Short	No	H
6	White	2	Tall	No	H
7	White	2	Tall	No	H
8	White	2	Short	Yes	H

New Instance

(Color=Green, legs=2, Height=Tall, and Smelly=No)

NAIVE BAYES CLASSIFIER

EXAMPLE - 2

$$P(M) = \frac{4}{8} = 0.5 \quad P(H) = \frac{4}{8} = 0.5$$

Color	M	H
White	2/4	3/4
Green	2/4	1/4

Legs	M	H
2	1/4	4/4
3	3/4	0/4

Height	M	H
Tall	3/4	2/4
Short	1/4	2/4

Smelly	M	H
Yes	3/4	1/4
No	1/4	3/4

Example : Naïve Bayes ML

From the below we understand that the new instance to classified as H is higher than M , so the new instance is H ,

$$P(M) = \frac{4}{8} = 0.5 \quad P(H) = \frac{4}{8} = 0.5$$

Color	M	H
White	2/4	3/4
Green	2/4	1/4

Legs	M	H
2	1/4	4/4
3	3/4	0/4

Height	M	H
Tall	3/4	2/4
Short	1/4	2/4

Smelly	M	H
Yes	3/4	1/4
No	1/4	3/4

$$p(M|New\ Instance) = p(M) * p(Color = Green|M) * p(Legs = 2|M) * p(Height = tall|M) * p(Smelly = no\ |M)$$

$$p(M|New\ Instance) = 0.5 * \frac{2}{4} * \frac{1}{4} * \frac{3}{4} * \frac{1}{4} = 0.0117$$

$$p(H|New\ Instance) = p(H) * p(Color = Green|H) * p(Legs = 2|H) * p(Height = tall|H) * p(Smelly = no\ |H)$$

$$p(H|New\ Instance) = 0.5 * \frac{1}{4} * \frac{4}{4} * \frac{2}{4} * \frac{3}{4} = 0.047$$

Example : Naïve Bayes ML

Advantage

- Simple
- Incremental learning
- Naturally a probability estimator
- Easily handles missing values

Disadvantage / Weakness

- Independence assumption
- Categorical/discrete attributes
- Sensitive to missing values

Example : Naïve Bayes ML Python Implementation

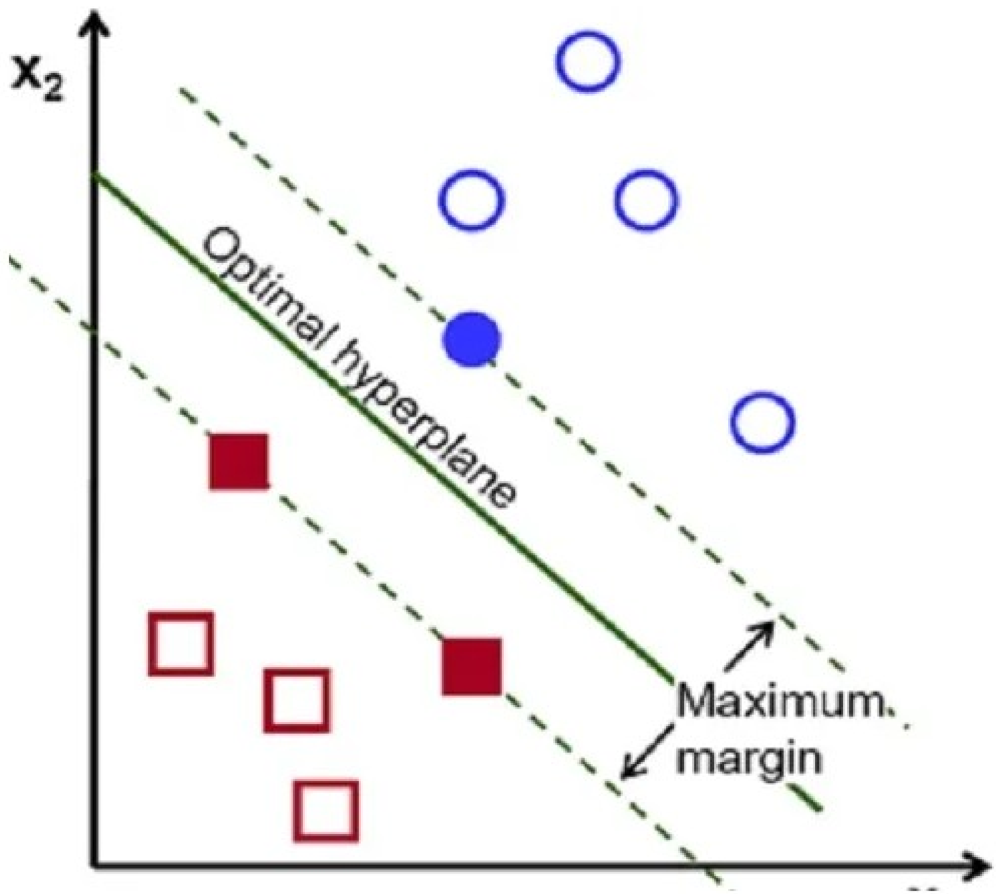
from sklearn.naive_bayes import BernoulliNB

```
# Initialize the Bernoulli Naive Bayes classifier
nb_classifier = BernoulliNB()

# Train the classifier
nb_classifier.fit(X_train, y_train)

# Make predictions on the test set
y_pred = nb_classifier.predict(X_test)
```

SVM Machine Learning algorithm



SVM Machine Learning algorithm

Support Vector Machine (SVM) is one of the **most useful supervised ML** algorithms.

It can be used for both classification and regression tasks.

Basic idea of support vector machines:

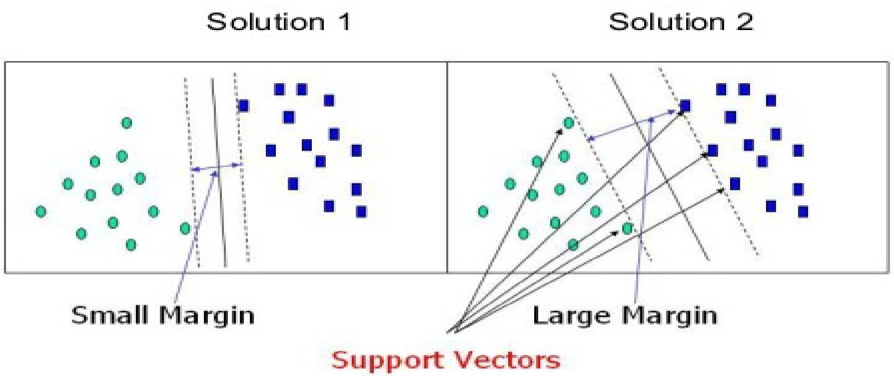
SVM is a geometric model that views the input data as two **sets of vectors** in an n-dimensional space.

- It constructs **a separating hyperplane in that space**, one which **maximizes the margin between the two data sets.**

SVM Machine Learning algorithm

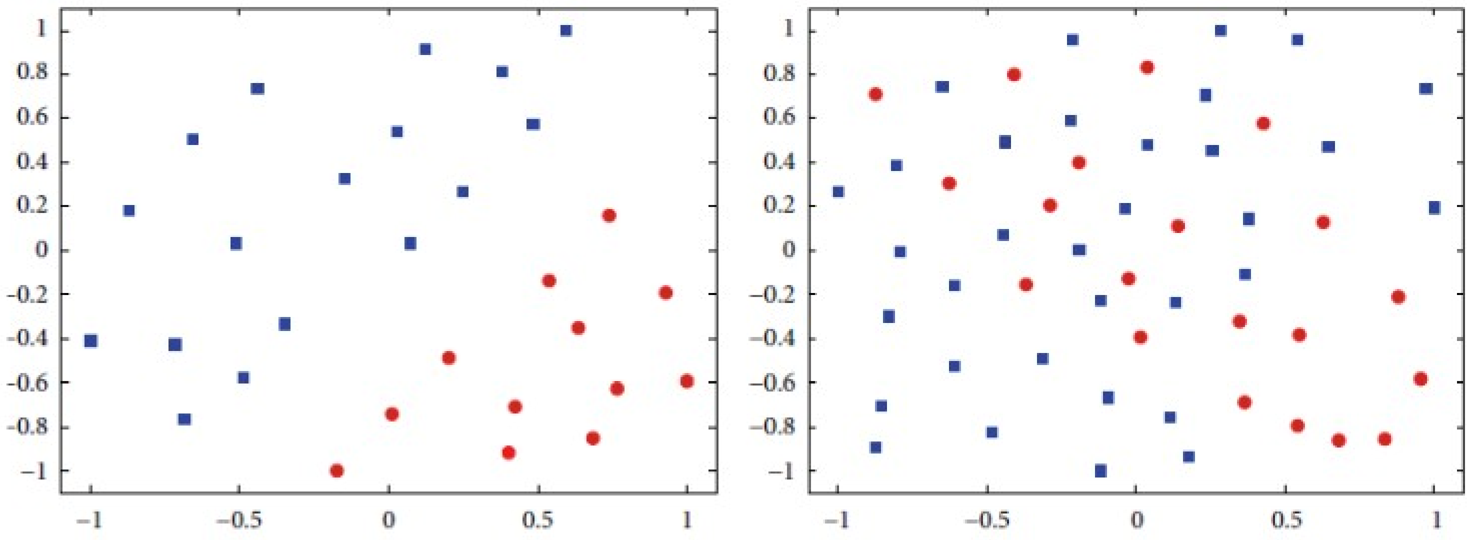
A good separation is achieved by the **hyperplane** that has the **largest distance** to the neighbouring data points of both classes.

- **The vectors (points)** that **constrain the width of the margin** are the support vectors.
- **Support vectors** are the data points that **lie closest to the decision surface**



An SVM analysis finds the line (or, in general, hyperplane) that is oriented so that the **margin between the support vectors is maximized.**
In the figure above, Solution 2 is superior to Solution 1 because it **has a larger margin.**

SVM Machine Learning algorithm

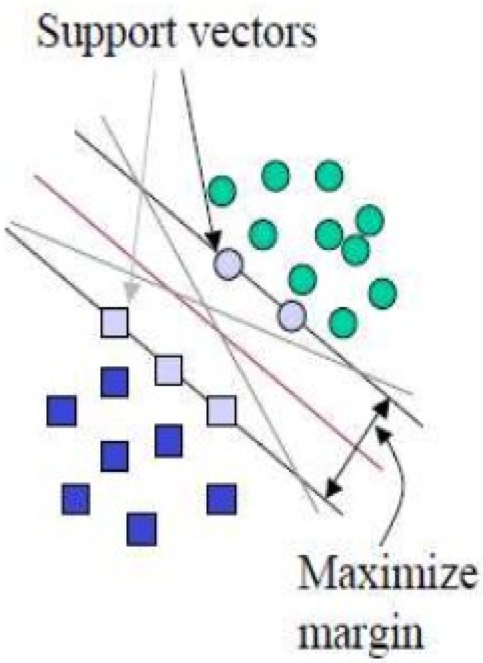


Which one is easy to separate?

SVM Machine Learning algorithm

SVMs maximize the **margin around the separating hyperplane.**

- The decision function is fully specified by a subset of training samples, the support vectors.
- 2-Ds, it's a **line.**
- 3-Ds, it's a **plane.**
- In more dimensions, call it a hyperplane.



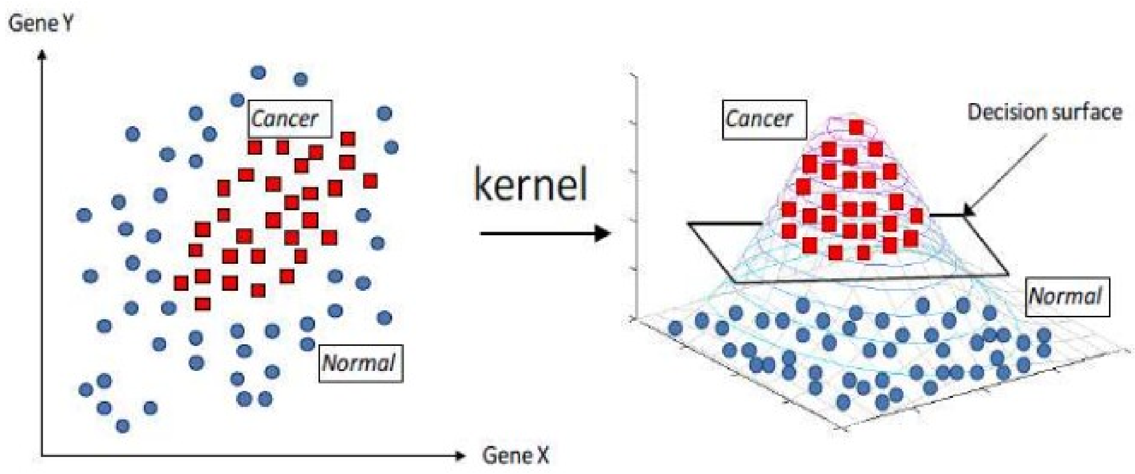
SVM Machine Learning algorithm

Basic idea of support vector machines:

– **hyperplane** for linearly separable patterns

-A hyperplane is a **linear decision surface** that splits the space into two parts

– For non-linearly separable data-- t**ransformations of original data to map** into new space – the Kernel function



SVM Machine Learning algorithm

Important because of:

– Robust to very large number of **variables and small samples**

– Can learn both simple and highly complex classification models

– Employ sophisticated mathematical principles to **avoid overfitting**

– Can be used for **both classification and regression** tasks

-Effective in cases of limited data.

SVM Implementation Python

Popular kernel functions in SVM:

The SVM kernel is a function that takes **low-dimensional input space** and **transforms it into higher-dimensional space**, ie it converts **nonseparable problems** to **separable problems.**

-

SVM Implementation Python

Scenario

Worldwide, breast cancer is the most common type of cancer in women and the second highest in terms of mortality rates. Diagnosis of breast cancer is performed when an abnormal lump is found (from self-examination or x-ray) or a tiny speck of calcium is seen (on an x-ray).

After a suspicious lump is found, the doctor will **conduct a diagnosis to determine** whether it is cancerous or not

Unsupervised Learning

- Unsupervised learning aims to find the **underlying structure or the distribution** of data. We want to explore the data to find some intrinsic structures in them.

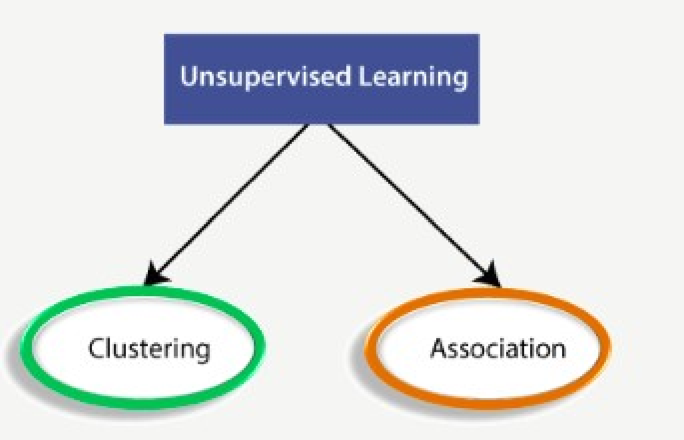
- unsupervised learning is a machine learning technique in which **models are not supervised using training dataset.**

- models itself find the **hidden patterns and insights** from the given data.

- Unsupervised learning **cannot be directly applied** to a regression or classification problem
- Unsupervised learning is much similar as a human learns to think by their own experiences, which makes it closer to the real AI.

- Unsupervised learning works on **unlabeled and uncategorized** data which make unsupervised learning more important.

Basic Steps In ML



Clustering: Clustering is a method of grouping the objects into clusters such that objects with most similarities remains into a group.

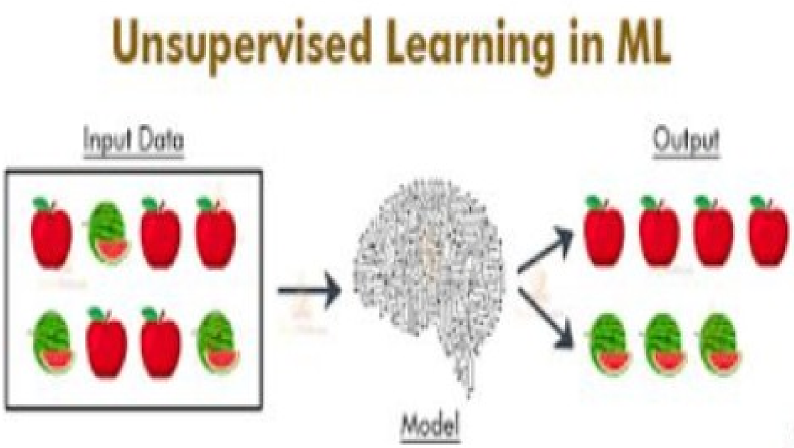
Association: An association rule is an unsupervised learning method which is used for **finding the relationships between variables** in the large database. It determines the set of **items that occurs together** in the dataset.

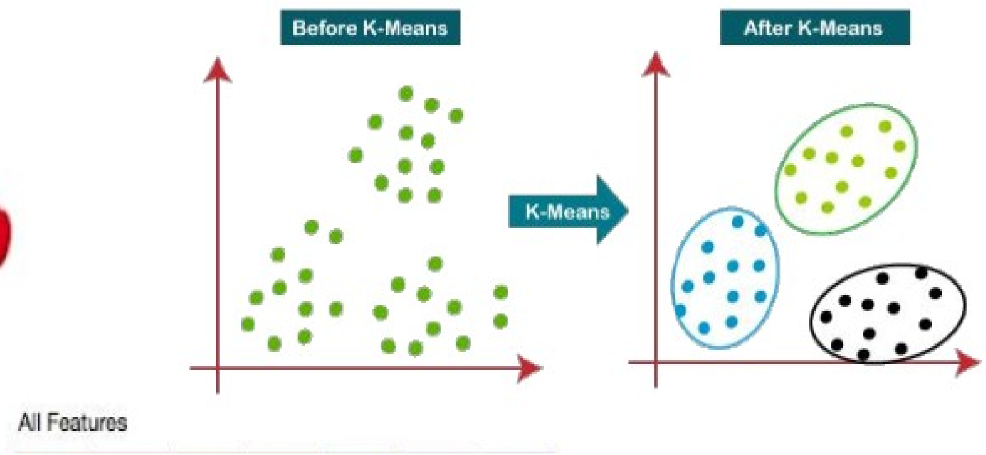
Unsupervised Learning algorithms

Below is the list of some popular unsupervised learning algorithms:

- ➔ K-means clustering
- ➔ KNN (k-nearest Neighbors)
- ➔ Principal component analysis

K-means clustering





K-means Algorithm

- **Definition:** K-Means is a **partitioning clustering algorithm** that separates a dataset into K distinct, non-overlapping subsets (clusters).

- **Working Principle**

- **Initialization:** Randomly select K data points as initial cluster centers.

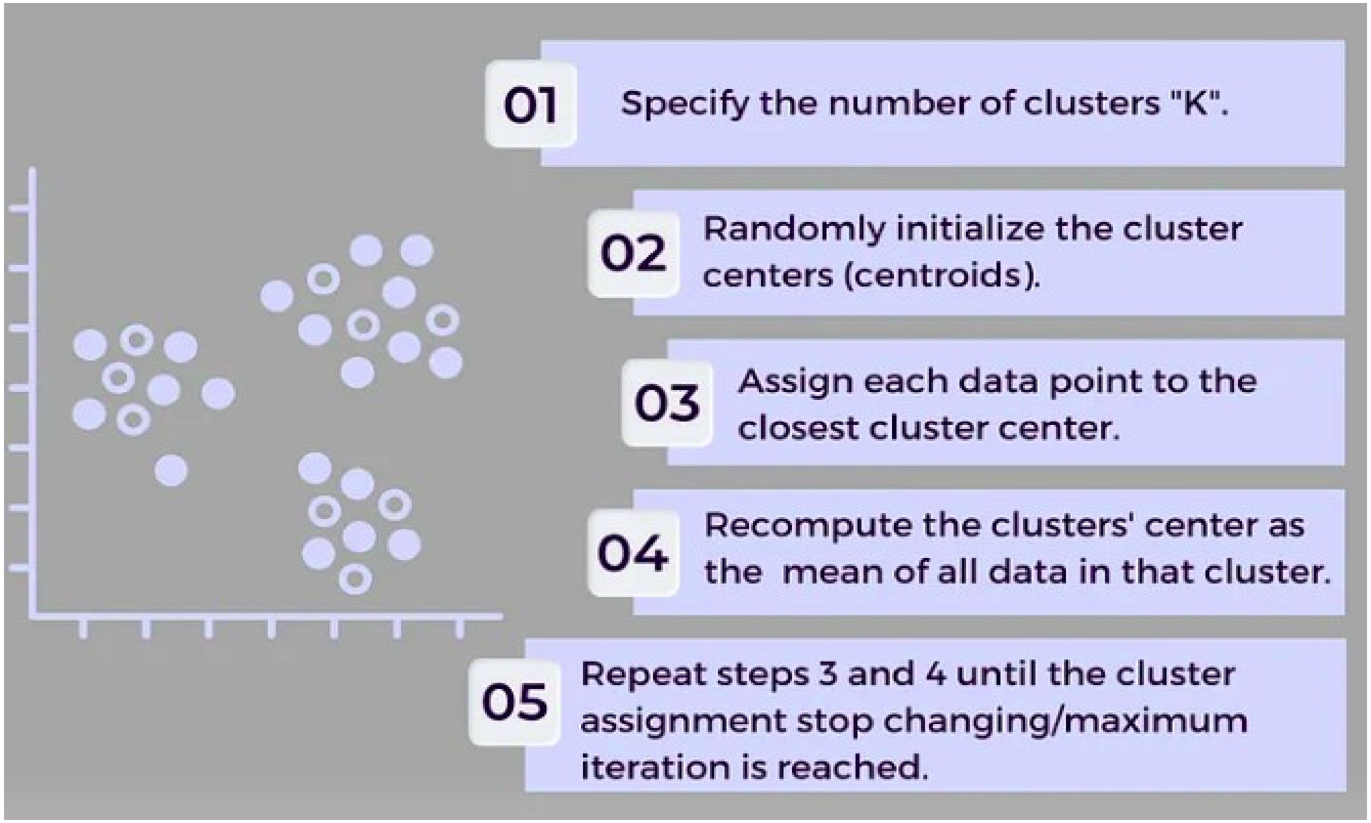
- **Assignment:** Assign each data point to the cluster whose center is nearest.

- **Update Centers:** Recalculate the cluster centers as the mean, variance, Euclidian distance of the data points in each cluster.

- **Repeat:** Iterate steps 2 and 3 until convergence (when cluster assignments stabilize).

Common Terms

- Given k, the k-means algorithm is **implemented in 5 steps:**



https://domino.ai/blog/getting-started-with-k-means-clustering-in-python

Python Implementation

- **Problem Statement:**

- A retail store wants to get **insights about its customers.** And then build a system that can cluster customers into different groups.

- **https://github.com/NelakurthiSudheer/Mall-Customers-Segmentation/blob/main/Python%20Code/Implemenation%20in%20Python.ipynb**